University of Glasgow | School of Computing Science

Travel Guide System

Dina Alsharif

School of Computing Science
Sir Alwyn Williams Building
University of Glasgow
G12 8QQ

A dissertation presented in part fulfilment of the requirements of the Degree of Master of Science at The University of Glasgow

Monday 13th of December 2021

Abstract

Sharing economy platforms in the hospitality field, such as Airbnb, provide the user with helpful information about available properties to stay in, such as its location, price, and host. However, they are not providing information about the nearby places such as cafes, restaurants, activities and transportation, or information about how likely the crime occurs in the property's neighbourhood. This information can effectively support the user's decision of selecting a place to stay. Thus, this project provides valuable information to explore property more in-depth. The Travel Guide application lets the user know how often the crime occurs in a neighbourhood, information about the property, its host, and summary of customers' reviews . Moreover, this application lets users know the nearby food venues, activities and transportation. In addition, the application provides information about valuable locations like embassies. This application works with different data sources, the Places API to provide information about the nearby places and transportation, DBPedia API to provide information about activity venues, and the last source is the database, which contains a large size of data that combined from different sources. These sources are used to overcome the information shortage in Airbnb and other platforms.

keywords: nearby venues, food venues, activities, transportation, property, Places API, DBPedia API.

Education Use Consent

I hereby give my permission for this project to be shown to other University of Glasgow students and to be distributed in an electronic form.

Name: Dina Alsharif Signature: DinaKS

Acknowledgements

I would like to thank you to my supervisor, Richard Mccreadie, for guiding and providing me feedback throughout the process of completing the dissertation. I would also like to thank all the participants who evaluate the Travel Guide application. They provided me with data that helped me to assess my final product and set plans for future work. Finally, many thanks to my parents who support me and help me to stay focused and motivated.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

In the last few years, sharing economy platforms have become popular in different fields, especially in hospitality [1]. These platforms enable cutting down the middle man in the supply chain process and making transactions directly between the supplier and customer (peer-to-peer) that results in providing products/services with low prices and convenience, that a reason behind the popularity of these platforms[2]. Airbnb is one of the popular sharing economy platforms that operate in more than 200 countries [3]. It allows millions of people (hosts) to invest in unused spaces by renting them to millions of guests [4]. Airbnb reduces the cost of properties between 30% to 60% compared to hotels[5]. Furthermore, the customer can filter the properties based on the price range, location and a wide range selection of property types (entire apartment, entire home, in-suite bedroom). Moreover, Airbnb provides the user with more customization options enabling the user to find a property that meets all the criteria that the user need. For example, the user could select a property based on the host language, interest or property amenities. Furthermore, Airbnb's revenue increases year by year. In 2018, the total revenue of Airbnb was 3.6 billion U.S dollars which increased by 31% in the following year[6].

1.2 Purpose

Even though the Airbnb platform provides helpful information about the property itself and its host, it does not provide information about nearby places (cafés, restaurants, public transportation, activities) or information about how often crime occurs in the neighborhood of the property. This information is important as it could strongly affect the user decision of selecting a place to stay.

This project aims to build a new web application that is going to solve these problems by creating a more in-depth property exploratory experience. In particular, this system is called Travel Guide, it would enable the users to analyze properties, but also venues within London, giving them more historical information about those properties as well as being able to explore different activates or venues that exist nearby. It would also provide overlies of information that allows the user to have broader information about the city such as crime rates, information about useful locations like embassies. In this way, the Travel Guide system will provide more effective information than Airbnb does currently that help the users to decide where to stay.

1.3 Outline

The project report is divided into five chapters to describe the life cycle stages of the system development. Chapter 2 focuses on the survey (related works and knowledge gab) and requirements analysis (functional/non-functional requirements and user scenarios), followed by Chapter 3, which mainly discusses design, including data flow diagram and implementation of the system. Then testing the system to find if the system meets the requirements or not in Chapter 4. Finally, Chapter 5 will deliver the conclusion of the project and additional suggestions for future work.

Chapter 2

Analysis and Requirements

This chapter will firstly, discuss the background knowledge of sharing economy platforms specifically property sharing platforms. Next, it will critically evaluate the existing products (related work) such as Airbnb and Booking.com, then compare between them to find the strength and weaknesses aspects in order to help fill the gap. After that, it will analyze the requirements by discussing user scenarios, functional and non-functional requirements and Moscow analysis.

2.1 Background

Since 2008, the sharing economy platforms in the hospitality field has been grown due to technology and economic development. Airbnb is a popular property sharing online marketplace that provides a new and affordable solution to satisfy traveler requirements for short-term stays [7]. As a part of the advantages, people have the flexibility to travel when they can stay in houses rather than hotels, however, due to the reasonable prices of accommodation in average compared to hotels people tend to stay longer time in the destination city [8].

In recent years, the number of Listings in Airbnb is more than 6 million[9]. Which means its market share is greater than the five largest hotel chains [9]. These millions of Listings allow Airbnb to provide a wide range of property types and unique experiences that could not be offered by hotels. To illustrate, Airbnb has yurts, treehouses, and boats to offer. The other benefit of a large number of properties is the user can customize the request, in other words, the users can easily find hosts that match their interests in a specific location [7].

Moreover, Airbnb customers are typically between 19 to 35 years old who heavily use the internet to purchase and prefer the convenience that comes with internet services [10]. An example of convenience in renting a property from Airbnb is that the price is fixed, the customer will not be asked to pay more for using equipment such as a washing machine or any service provided by the host such as Wi-Fi. In addition, “Belong Anywhere” is an attractive marking logo used by Airbnb that indicates customers can have a convenient local experience. In other words, hosts can provide the guest local food, dinner with local people or suggest some local activities [10].

2.2 Related Work

Airbnb, Housetrip, and Vrbo are property sharing web applications. Moreover,Booking.com not only provides user with hotels it also provides shared properties or entire places that supplied by individuals or companies. All these applications allow the user to select property based on its location and other criteria such as number of bedrooms, property type, facilities and price. These

applications provide a map that shows property location (not exact location for security purposes) unless the user booked it and markers on the map provide summary information about the property, such as image and price. Moreover, they provide detailed information about the property itself, the host, reviews and amenities. This information could strongly support the user decision in whether to book a specific property or not. However, notably, Airbnb does not allow the user to explore nearby places such as restaurants or public transportation.

Vrbo provides a list of nearby places. This list is general shows the most popular places in the city and the distance between the property and the suggested place. However, even though this suggested nearby places could help the user to explore the city, it lacks historical information about the place itself or provides the user with information about activities nearby. Moreover, the nearby places list does not cover restaurants and public transportation.

Booking.com provides the user with nearby places categorized into different categories (What's nearby, Top attractions, Restaurants and cafes, Natural beauty, public transport and Closest airports) and also provides the distance between the places and the property. However, the suggested nearby places lack information such as lines that are served by the nearby train station, cuisine type, meal or rating if the nearby place is a restaurant/cafe or provide the user with historical information about activities nearby.

Housetrip provides additional map within each property page shows the nearby places categorized into two categories (Attractions and Restaurants). It enriches the app with information about nearby places by linking each marker on the map with an associated Tripadvisor page, which facilitates the search process for the user. Moreover, the attractions category covers public transportations, activities and popular places.

Inside Airbnb, is web app provides a dashboard with a map that shows the distribution of properties within a city. This dashboard provides information about the number of each type of property, estimated data about host income, estimated occupancy, the ratio of review per listing per month, the number of listings are available all year round, number of listings with low availability. In addition, Inside Airbnb provides a list of top hosts and that could help the user in making his decision about where he will stay.

All the above applications provide the user with the essential information that supports the decision of where he can stay except Inside Airbnb. Additionally, they do not provide information about the crime rate in the property neighborhood, or information that helps the user to contact his embassy in extraordinary circumstances. In this project, I will build Travel Guide, a new application that aims to close the technology and knowledge gap by building a system that allows the user to filter properties based on location, type, number of bedrooms and price. Moreover, it will provide a map with markers for each property that provide a pop-up summary about it, such as property name, image, price, number of bedrooms and accommodates number. Not only that but it will also provide crime rates in neighborhoods, the contact information of embassies, information about properties and their hosts and nearby places (restaurants, cafes, public transportation, and activities) with detailed information. The detailed information means that the user can know if the nearby restaurant/café provides meals for breakfast, lunch or dinner also the user can filter the restaurants/cafés based of diet type (vegan, vegetarian, gluten-free). The application provides information about cuisines, address, distance and rating. Moreover, the system will provide the user with a reviews summary by analyzing all the reviews that written by customers for each property to identify customer sentiment, then display it in a bar chart (negative, positive and nature). That will help to save users time instead of reading all or most of the reviews to know other people's opinions.

2.3 Requirements Analysis

2.3.1 User Stories

Tala is 35 years old. She likes to travel to live new experiences in new countries by renting a property from local people, either shared or non-shared. Therefore, Tala always uses the Airbnb application to rent a property. However, she has an issue every time before booking a property; she needs to check the property location, the crime rate associated with that location, the nearby public transportation and cafes because Tala prefers to drink a cup of coffee in the morning once she wakes up.

John is a professor; he usually travels to attend conferences and workshops. Therefore, he needs to have enough time to sleep, and he does not want to waste his time travelling between the conference venue and the property. John prefers to rent a property near the conference venue. He needs to finds a property in the same neighbourhood of the conference venue and checks the nearby public transportation before booking. Moreover, some of John's friends are vegan, and they like to have a lunch meal together during the conference or workshop. Therefore, John needs to filter the nearby restaurants based on the diet type.

Dina is 22 years old; she likes to visit tourist-friendly cities with a low crime rate. She continuously checks the crime rate of the city neighbourhoods. Dina always wants to feel safe; therefore, she searches for her country embassy information and saves the contact detail in her mobile phone in case she encounters any problem, such as losing her passport. Moreover, she is organized, and she prefers to prepare herself depending on the city's available activities; therefore, she needs to know the type of activities. For example, if the activity is diving, she needs to have diving ware in advance.

2.3.2 Functional Requirements

• Find a place to stay in the city. The user can find properties in different locations in the city (London).

• Know if the neighbourhood of the property is safe or not safe. The user can designation between high, middle and low crime rates of neighbourhoods.

• Find public transportation (train or bus) nearby the property.

• Find a restaurant, café, and activity nearby the property. The user can find information about the nearby restaurant, café or activity.

• Find information about the user's country embassy. The user can find the contact information and location of his country embassy.

2.3.3 Non-functional Requirements

• Reliability, the application can work effectively anytime.

• Usability, the user can easily use the application.

2.3.4 Moscow Analysis

Must Have

• RQ1: User can find property based on the location with detailed information about it and its host.

• RQ2: User can know the nearby restaurant/café and activities.

• RQ3: User can find the nearby public transportation.

• RQ4: User can know how likely the crime occur in a neighbourhood.

• RQ5: User can find information about his country's Embassy.

Should Have

• RQ6: User can filter the properties based on some criteria (bedroom number, property type, neighbourhood, price per night) .

• RQ7: User can know the distance between the property and nearby public transportation, restaurant, café and activities.

• RQ8: User can know the summary of property reviews (sentiment analysis).

Could Have

• Recommend host to the user based on matched interests.

• Predict the property price in near future.

Won't Have

• Filter properties based on the host's language/s.

• Discount codes for restaurant/café nearby the property.

Chapter 3

Design and Implementation

This chapter will discuss the design and implementation in detail. The first section will display the system architecture using a level-0 data flow diagram to introduce the general idea of the application, and the data flow diagram to show how the data move from its source to its destination (web-page). After that, this chapter will discuss the technologies that are used in order to build each requirement in sections 2.3.4. Furthermore, it will list all the data sources that were investigated either used or not used in the project. The last section of this chapter will show the application implementation.

In analyzing the related work in Section 2.2, it was found that all the existing applications are websites. Therefore, the Travel Guide app will be a web application for two main reasons. The first reason is "consistency" that the users are familiar to use web apps while they are looking for a place to stay. The second reason is that web apps are not required to install on a local machine, so the user can access it from any computer device with internet connection. This project is managed by using the waterfall approach, this approach is a combination of sequential stages starting with requirements analysis then, design, implementation after that testing [11]. However, this approach was selected because the project has a fixed time and the scope is static.

3.1 System Architecture

The Travel guide system is considered as data driven system which combines different data sources in order to provide the users with information that supports their decisions. Therefore, the level-0 data flow diagram or context diagram is selected to display the entire system as a single process and focuses on the input data source and the output information [12]. This diagram clarifies the main/general idea of the system.

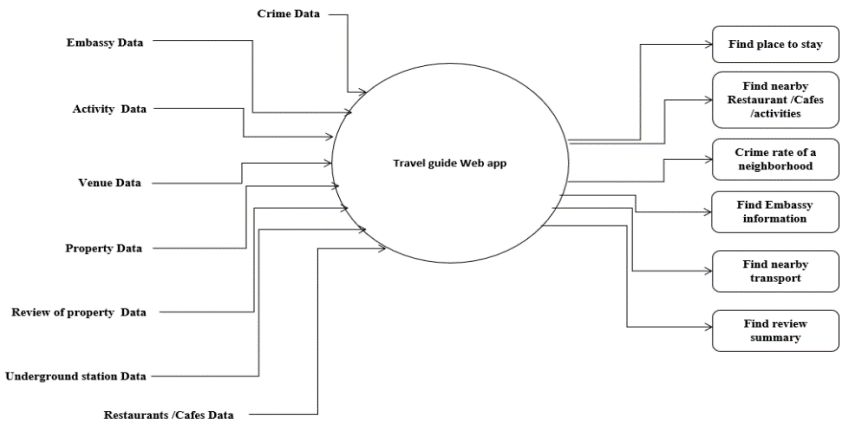


Figure 3.1: Level-0 data flow diagram.

The below data flow diagram shows the pipeline of the data includes the source files, APIs and the way data moves from these sources to their destinations in the web app.

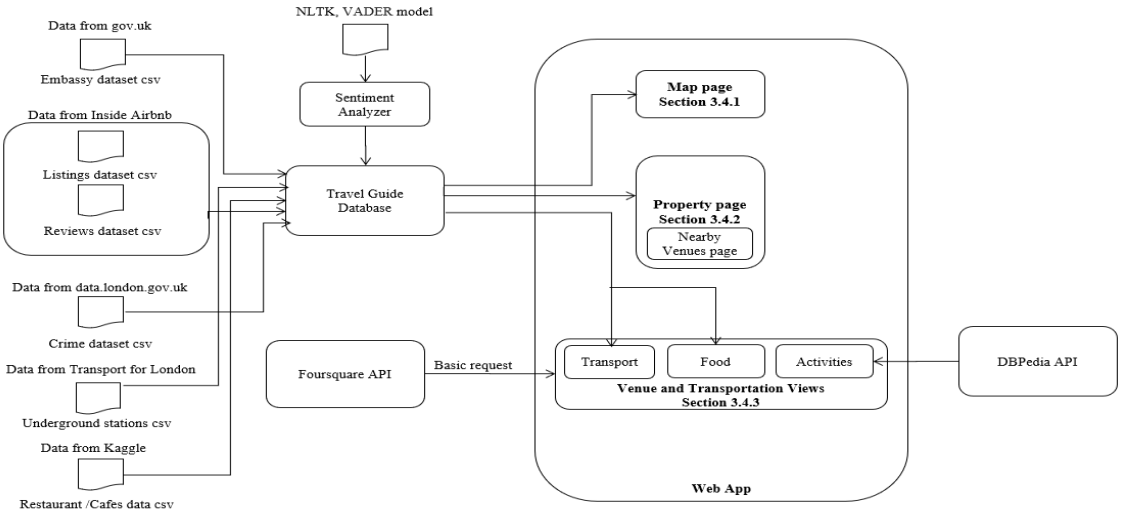


Figure 3.2: Data flow diagram.

3.2 Technologies

In order to build this data-driven web app, multiple technologies are used to process the data (analyze, clean, aggregate, display). The primary programming language is Python and it was selected for many reasons. The first reason is Python is a vector-based language which means developers can perform mathematical calculations on the entire column (vector) as it is a single object without the need to loop over each index [13]. The second reason is that it is designed to be used by data scientists which provide them with multiple built-in functions such as Numpy (provides mathematical functions), Pandas (Deals with data frames) and function of text sentiment analysis that is used in the Travel guide app to analyze the reviews [14].

The Colab platform is used in the data engineering process (cleaning data) because the size of the raw data was huge. Therefore, they were uploaded to Google Drive, which allows the developer to store up to 100 GB with a basic plan, Python notebook on Colab can access and manipulate the data stored in Google Drive by using drive.mount() [15]. The Colab was not used to develop the Travel Guide application, and that is because, when the Map page was developing there, it was one challenge that stuck the developer to continue the development with the Colab. The challenge was, when the Dash application is run in one local port, this port will be in use all the time and will not allow the application to re-run again using that port. To illustrate, if the developer wants to check the output of the code, it will be run for the first time or sometimes will run for the second time, but then it throws an error. Even when the developer tries to terminate the session and re-run the application, the system will throw an error of "Address already in use". Alternatively, PyCharm is a Python IDE that was used to build the Travel Guide application [16]. After cleaning the data, its size was reasonable to store in a local machine.

The Travel Guide app uses VADER model from the NLTK package to analyze the emotions of unlabeled text and classify it into three classes (positive, negative and neutral), this type of classification is called polarity classification [17].The VADER model is used because it is efficient to analyze text and classifying it. This model is used to meet RQ8, which is provides the user with summary of property reviews.

The data can be visualized by using an interactive dashboard which is a library in python that is called "Dash". Dash app run on web-page easily, it has two main components layout and callback. The layout is the front-end part where the page design and style are built, and the callback could be considered as the back-end where the functions/operation are located, specifically, the definition of the callback functions is "functions that are automatically called by Dash whenever an input component's property changes, in order to update some property in another component (the output)" [18].Therefore, callbacks are used instead of classes. Moreover, the app style is created by using Bootstrap which is an open-source library that provides different style options such as colour s, buttons and text style. Bootstrap is facilitating and speedup the process of styling the HTML elements in a web page[19]. Folium from the leaflet library is used to integrate OpenStreetMap in the dashboard and draw the boundaries of London's neighbourhoods on the map. The combination of Dash and OpenStreetMap are used to meet RQ1 (Find property), RQ4 (crime rate), RQ5 (Embassy information) and RQ6(filter properties).

Sqlite3 is a library that allows the system to interact with a locally stored database. However, to extract and deal with external data sources such as venue data and information about activities the application uses APIs of Foursquare Places and DBPedia. The application uses specifically sandbox API from Foursquare with explore endpoint it has a limit of daily calls which is 950 for regular calls and 50 for premium calls, this number of calls is sufficient for the Travel guide app. [20] Moreover, DBpedia API allows the application to extract data from Wikipedia articles, and the endpoint is (resources/venue name) to extract information about specific activity places. APIs of Foursquare and DBPedia are used to meet RQ2 (Find nearby restaurants/cafes and activities), RQ3 (Find nearby public transportation) and RQ7(calculate the distance between property and suggested places).

To integrate data from different sources to serve one destination(page), to illustrate, for example, the food venue pages are supplied with data from Foursquare places API and data stored locally in the database in a table called London_restaurants. In order to combine these data, the Travel Guide application uses the FuzzyWuzzy library, which calculates the similarity between text data to find the similarity between data that come from two sources, then the application can integrate them (see

Section 3.4.3) [21].

3.3 Data Collection, Extraction and Storage

This project is a data-driven system (dashboard), so it aims to use real data. The data collection and extraction phase was the first step after the project planning process. During the data collection process, the developer collected raw data in different formats and manipulated them to extract, uniform, aggregate, analyze and store clean data. When the data become clean, it will save the cost of calculation and storage space [22].

3.3.1 Data Sources Investigated but not Used

Accessing data was one of the challenges that the developer encountered with data providers having their own policies and restrictions. The cost of using the data is another challenge. Moreover, issues related to data access are restricted by the policies and restricted by its price per call.

Tripadvisor: The developer initially intended to use data from the Tripadvisor, which is one of the largest platforms that allows the user to generate content in the hospitality field, such as writing a review about restaurant or activity venues, to provide the user with more information, images and review summary (sentiment analysis) for nearby food and activity venues to meet RQ2. [23] However, when the developer contacted Tripadvisor asking for accessing their data, unfortunately, their data are not available for individuals, it is only accessible for organizations.

Google Maps Places APIs: The developer also encounter an issue to use the Google Maps Places API, which is a source that provides information about places using HTTP requests, the aim of using it was to provide the user with information about the nearby venues to meet RQ2, moreover, the distance between selected property and nearby venues to meet RQ7 [24]. However, the data that comes from Places API is not free, the price varies from about $3 upto 30$ per 1000 usage [25].

3.3.2 Data Sources Used

To overcome the issues associated with data collection, the developer extracted the data from seven free of charge sources and had fewer usage restrictions. The raw data files were huge, and the developer needed to store them in Google Drive instead of storing them in her local machine. These data were processed in a separate Jupyter notebook, specifically using colab platform, where colab allows the developer to process and read the data stored in Google Drive by using drive.mount(). To illustrate, The raw data of Customer's reviews contain more than ten million records. After cleaning and processing them, they are stored in the listing table in the database, thus, the number of records was exponentially decreased to around 65,000 records. However, all the collected data were about only one city, which is London in the UK, due to the scope of this project.

Embassy Data: this data comes from GOV.UK specifically, from the London Diplomatic List file. The issue of this source is the data stored in an unstructured format (text file), the developer enforced to copy/paste all embassy information to CSV file. Therefore, the CSV file does not contain all the embassies, it only contains a few. However, this CSV file is then stored in the database in a table called the embassy. This table contains these fields country, address, contact_number, email, website, and Embassy_name. This source allows the application to meet RQ5.

Listing Data: this data is about Airbnb Listing comes in two files. The first file is called Listings, and the anther one is called reviews, they are provided by Inside Airbnb. The developer com-

bined data of six months for both Listings and reviews. The combined Listings data are stored in the database in a table called Listing (Listing_url, name, description, neighborhood_overview, picture_url, host_about, neighbourhood, neighbourhood_cleansed, latitude, longitude, property_type, accommodates, price, review_scores_location) to meet RQ1and RQ6. Moreover, all the reviews were processed by the NLTK package to analyze the emotions associated with customers' reviews and classify them into three classes (positive, negative and neutral) to meet RQ8. These three classes are stored in three columns (pos, neg, and neu) in the database in the Listing table.

Crime Data: this data is about the number of crimes that occurred in the London neighbourhoods from January 2011 to August 2021, the source of this data is London Datastore. This source is used to meet RQ4. The developer calculated the average crime number per month per neighbourhood, then generated the latitude and longitude for each neighbourhood in this data file. After that created a table in the database is called crime to store the neighbourhood name, latitude, longitude, and average crime number. However, this table was created to visualize the crime rate on the map by using choropleth, but this was not implemented as the developer expected (see Section 3.4.1 for more information). The crime table is kept in the database for future enhancement. To overcome the issue of visualizing crime rate with choropleth layer, the developer divided the neighbourhoods into three categories (low, middle and high crime rate) by using the interquartile range (see Section 3.4.1 for more information) each category is associated with colour . Where green indicates low level, orange for middle level, and red indicates high crime rate after processing the data and linking crime rates with Listings based on the neighbourhoods, each property is associated with a colour stored in the colour column in the Listing table in the database.

Venues and Transportation Data Sources There is one source supply different data to different destinations (pages) that is Places API from Foursquare(see 3.2), which is an application that provides information about places nearby the user. This source provides data for food, activities and transport pages using the "explore" endpoint. In addition, the developer was using a sandbox account. This type of accounts is free of charge and allows the application to use 950 for regular calls and 50 for premium calls [20].

Food Venues Data Sources: pages of Breakfast, Lunch, Dinner and Café are supplied with data from two different sources to meet RQ2 and RQ7. The first source is Place API (explore endpoint) from Foursquare that provides information of venue name, the distance between the selected property and the venue, venue category and venue location. The second source from Kaggle about European restaurants, the developer extracted only venues in London and stored them in the database in a table called London_restaurants. This table contains restaurant_name, original_location, region, province, city, address, latitude, longitude, meals, cuisines, special_diets, Vegetarian, Vegan, Gluten_free, avg_rating. The main reason for using the second source is to allow the user to filter the food venues based on the diet types.

Transportation Data Sources: pages of Bus and Train are supplied with data from Foursquare Places API. The bus page uses only one source to provide it with the bus stop name, the distance between the selected property and nearby bus stops, its addresses and category. However, the train page uses two sources the first one, as machined before, is Foursquare Places API to provide it with the train station name, the distance between the property and the train station, its address and category. The second source is Transport for London that comes in a pdf file converted to a CSV file, and some merge and aggregation operations are applied to it. When it became clean, the developer stored it in the database in a table called underground_stations that contains only two columns, station and line. The purpose of using the second data source is to let the user know which lines are served by the nearby train station. However, these data sources are used to meet RQ3 and

RQ7.

Activity Data Sources: activity page utilizes two sources of data as shown in figure 3.2. The first source is Foursquare Places API to provide the user with the venue name, the distance between the venue and the selected property, venue category and location. The second source is DBpedia API is used to provide the user with information about the activity venue. These sources are used to meet RQ2 and RQ7.

Travel Guide Database: the database was used to store the cleaned data from the sources discussed above. However, the database contains five tables. The main table is **Listing** that contains data about property and crime rates, where each property record is associated with a colour that indicates the level of crime in the neighbourhood. Moreover, this table contains information about properties, their summary information of reviews (positive, negative and neuter), and their hosts. **The Crime table** contains information about the average crime number per month per neighbourhood, this table was created in order to build a choropleth layer to visualize the crime rates on the map but this could not be done as expected, for more information see Section 3.4.1. this table is kept in the database for future enhancement. The relationship between the Listing and Crime tables is many to one, which means many properties in the Listing table belong to one neighbourhood (SNTBoroughName) in the Crime table. **London_restaurants** is a table that provides information about food venues and stores data that allows the user to filter the food venues based on the meals and diet types. However, the relationship between London_restaurants is many to many, which means many restaurants are near many properties, the links between these tables are the latitude and longitude. **Embassy** table was created to provide the user with contact information of embassies in London. The last table is **Underground_stations** that supplies the application with underground lines that are served by each train station in London. See C.1

3.4 Implementation

This section will introduce the challenges and solutions that occurred during the project implementation. This section is divided into three parts, the first part discusses the Map page development iterations, challenges and solutions. The second part will present how the Property page was implemented, and the last part going to show Venues and Transportation pages implementation.

3.4.1 Map Page

The map page is the main page of the Travel Guide application and it allows the application to meet the requirement of RQ1, RQ4, RQ5 and RQ6. When the user opens the application, it will be the first page the user can interact with.

The first version of this page was built by using a scatter plot on a Mapbox. It was hard to build two layers on the map to show the crime rate on a layer underneath the dots, so the application used a scatterplot to colour the dots based on the crime rate, and the map was basic without pop-ups. However, after receiving feedback, the leaflet map was used to build a better map. (See figure A.1)

The final version of the map page is divided into two columns in the first column provides the user with five dropdown lists to filter properties on the map (RQ6) and the second column contains the map (RQ1) as shown in the figure 3.3. The first dropdown list allows the user to select neighbourhoods he would like to stay in, the second dropdown list allows the user to select the type of the property, the third one is to select a number of bedrooms, the fourth and fifth ones to set the price

range (minimum and maximum). To illustrate, see figure A.2.The system will show an error message when the user insert minimum price greater than the maximum price and will not show any property marker on the map as shown in the figure A.4

Figure 3.3: Map Page.

After the user selects the criteria, the filtered property marks will appear on the map with three different colours, the colour code is used to indicate the level of crime rate in different neighbourhoods (RQ1 and RQ4) where green means the crime rate is low, orange means middle and red means high crime rate. Each marker on the map is associated with a pop-up that shows summary information about the property, the Listing name, accommodates, number of bedrooms, price, image and its URL (RQ1) as shown in the figure 3.3.

The main technology used to build the dropdown lists and link them to the map is a library called Dash. Folium is a library that allows the application to visualize Listing data into an interactive leaflet map (open street map). This library allows the developer to build two layers on the map (RQ1).

The first layer is the choropleth, this layer was initially built to present the London map with a colour scale (light colour indicates a low crime rate, dark colour indicates high crime rate, and the colour between light and dark indicates the middle level of crime). Even though the developer created the crime_scale object with numpy.linspace to build this layer to visualize the crime level in different neighbourhoods, it did not result in the intended output. However, that could be because the "City of London" 's geographical boundaries are not included in the geojson file. In other words, not all crime locations match geographical data from (London_geo is a geojson file). The benefit of using choropleth is to visualize London's neighbourhood boundaries, which allows the user to expect to see property markers within the shadow area.

To overcome the issue of visualizing the crime rate on the map and display where exactly the property is located, the second layer (marker) is used (RQ1 and RQ4). The developer decided to categorize neighbourhoods into three categories. To do that, the developer calculated the crime average per month per neighbourhood, then these neighbourhoods were divided into three categories (low, middle, and high crime rates) based on the interquartile range (IQR), this measure is used to

divide the data into quartiles where Q1 is the middle value of the first half of the ordered dataset, Q2 is the median of the dataset and Q3 is the middle value of the second half of the ordered dataset [26]. After looking into the summary statistics, Q1 was about 163 and Q2 was about 208. Low crime rate (green markers) was assigned to neighbourhoods with a crime average is less than Q1, Middle crime rate (orange markers) was assigned to neighbourhoods with a crime average between Q1 and Q2, and the high crime rate (red markers) was assigned to neighbourhoods with crime average greater than Q2.

One more dropdown list in the first column allows the user to select his country name to display information about the country's embassy, Embassy name, contact number, email and website (RQ5) as shown in the figures 3.3 and A.5.

The Embassy section was built by using Dash. However, there are only a few embassies in Travel Guide and that because the embassy data comes in unstructured data format (word file), which is difficult to extract, the developer ended up copying/pasting them in a CSV file then stored it in the database in a table called embassy.

3.4.2 Property Page

The user will be navigated to the property page when he clicks on the Listing link in the pop-up. as shown in the figure 3.4, this page provides the user with detailed information about the property, its host, neighbourhood overview, location, rating, button to navigate the user to Airbnb to book the property, bar chart that presents a summary of customer sentiment in reviews (negative, positive and nature). Moreover, it provides the user with nearby places for breakfast, lunch, dinner, cafes and activities and nearby train and buses. This page is mainly built to meet RQ1, RQ2, RQ3 and RQ8. The property page uses only the data that comes from inside Airbnb.

However, the property page uses Dash specifically card component. The card is a content container, which helps build the property page and link it with the map page. The developer tried to build it as a basic HTML page with Flask but this option was not able to link with the map page, and that because the map page is complex it was built with multiple layers. The first layer is the dash then the second layer is the map with the other two layers as described in Section 3.4.1.

Figure 3.4: Property Page

Property Information: this section was built to meet RQ1. This section provides information about the property by using data from Inside Airbnb that stored in the Listing table in the database. Moreover, the user can know the property name, its page in the Airbnb, description about it, its location, neighbourhood overview, information about host, and rating. See figure 3.4.

Review Sentiment: to build the bar chart that presents data about customer sentiment in reviews (RQ8), it is required to analyze the reviews using the VADER model from the NLTK package [17]. Analyzing and classifying the reviews into three categories (positive, negative and neutral) was done offline and stored the results in the database specifically, in the Listing table to make the navigation process to the property page faster than if the analyzing process was done online.

Property Image: the property image position could not be moved to the top, it stays in the middle of the column even when the developer set the top parameter of the dbc.CardImg equals to true or puts dbc.CardImg object inside a dbc.Row and change its position to the top, the image position still as is. However, the purpose of using the image int the property page was to meet RQ1.

Nearby Places and Public Transportation: this section of the page provides different filtering controls (buttons) that will allow the user to navigate to the next pages which are venues and public transportation (RQ2 and RQ3). When the user clicks on one of these buttons some data will be passed from the property page to the nearby venues or transportation these data are property latitude and longitude, moreover, the query that specifies the venue category. For example, if the user clicks on the Breakfast button, the data that passed is ”Breakfast” then the query value will be ”Breakfast”, and the latitude, longitude will be set as values of the ”ll” parameter, this parameter will use in the GET request that sends to the Places API in order to get venues that are filtered based on the query and nearby the property location. (See figure B.1).

3.4.3 Venue and Transportation Views

In this section, we will show four different views. Basically, all the venue pages use data that comes from Foursquare API, as shown in figure 3.2. Building an unknown length list of venues in the interface of the dash was one of the challenges that met the developer. The div object in the layout can only receive one list, to overcome this issue the developer stored venue information in a card, and then append each card in an empty list in the callback section and then store the large list in a list to return it to the div object in order to display them on the screen.

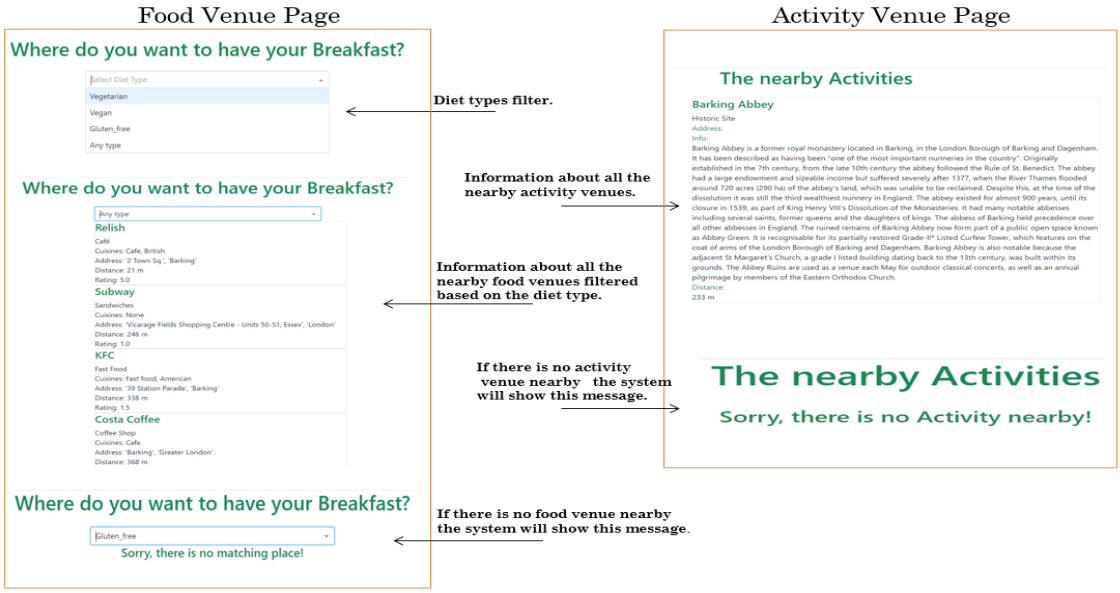


Figure 3.5: Food and activity venue pages

The first view is **food** which presents data about the nearby restaurants and cafes in order to meet RQ2 and RQ7. The user can navigate to this view when he clicks on the breakfast, lunch, dinner or café buttons on the property page. In these pages, the user can filter the restaurants and cafes based on the diet types (vegan, vegetarian and gluten-free) as shown in the figure 3.5. For this purpose, food venue pages use additional data source from Kaggle about European restaurants, the developer extracted only venues in London and stored them in the database. However, if the user does not want to filter the venues, he can select "any type" to show all the nearby restaurants or cafes. The application will provide the user with information about the venues (venue name, cuisines, address, distance between property and the venue, and rating) from the two data sources as shown in the figure 3.5. If there is no restaurants or cafes nearby matching the user's criteria the user will receive a message from the application that says "Sorry, there is no matching place!"as shown in the figure 3.5.

The data the comes from the two sources need to tie, however, even though both of them have venue names but the names were not matched due to spelling mistakes or different character cases. These sources also contain geo data (latitude and longitude) but it comes with different lengths of the decimal part so when the developer tried to round up the decimal part to uniform them, a new issue occurred that the location being inaccurate and it returns all the branches of one venue. To illustrate, Costa coffee has multiple branches in London when the decimal part of the latitude and the longitude were rounded up, the application returned all the branches for the user, and not all of

these branches are nearby the property.

To overcome this issue, a text-matching library was used, it is called FuzzyWuzzy. This library calculates the similarity between two strings [21]. The application used it to link the two data sources by calculating the similarity of venue name and address to return only venues nearby the property. However, the side effect of this solution is that the calculation takes a long time and that because the London_restaurants data is quite large, contains around 29,000 records. In other words, each venue name and address provided by Places API will be compared with all the records in the London_restaurants. See Section 4.2 for more information about loading time.

The second view is for **activities** is built in order to meet RQ2 and RQ7, this view/page aims to provide the user with the nearby activities with some detailed information (venue name, venue type, information about the place, and distance). If there is no nearby activity the application will tell the user "Sorry there is no activity nearby!" as shown in the figure 3.5. In order to provide the user with information about the activities, the application uses data from DBpedia API. To link the data from foursquare with the data from DBpedia the application uses the FuzzyWuzzy library to calculate the similarity between venue names in both sources.

The last views are for presenting information about nearby **transport** (trains and buses) in order to meet RQ3 and RQ7. If the user selects **train**, he will be provided with information about the nearby train station including station name, lines of trains, address and distance between the train station and the selected property as shown in the figure 3.6. However, if the application does not find any nearby station will show a message for the user saying that "Sorry there is no train station nearby" as shown in the figure 3.6. In order to provide the user with which lines are served by a train station, the application uses additional data source (Transport for London) this data is stored in the database specifically in underground_stations table. The station names from the foursquare and underground_stations table are not exactly the same and that is why the developer linked them by using the FuzzyWuzzy library.

However, the **bus** page provides information about the nearby bus stops that include (bus stop name, address and distance) as shown in the figure 3.6. If there is no bus stop near the property the application will show a message that says "sorry there is no bus stop nearby!". This page is sample uses only one data source which is Foursquare API and was built when the developer solved all the issues related to displaying the data on the screen in the previous pages, so the developer did not encounter any challenge during building the bus page.

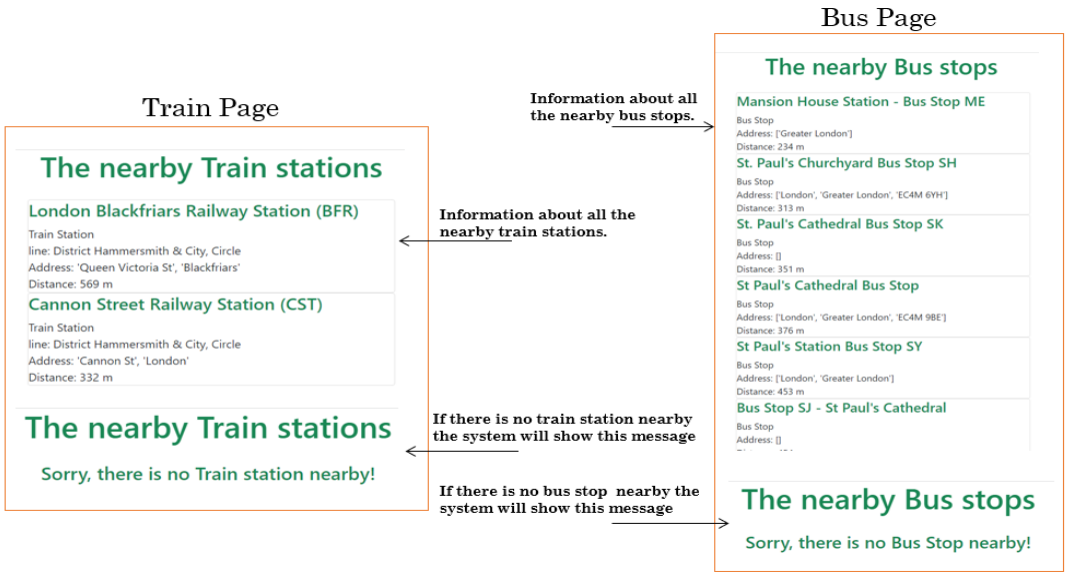


Figure 3.6: Train and bus pages

Chapter 4

Testing and Evaluation

This chapter will assess the project in terms of satisfying the functional requirements in Section 2.3.4 and non-functional requirements in Section 2.3.3. The first section will test each feature provided in the project divided into test cases. Next, the latency of each page in Travel Guide will be tested. After that, The user evaluation section allows real users to assess the project from their perspective and provide suggestions to improve it.

4.1 Test Cases

The test cases are created to test each function that Travel Guide provides, and see if the functions are work correctly or not. This section is divided into seven sections.

Test case ID	Test Case Description	Expected Results	Actual Results	RQ#
Test creating and updating property markers on the map				
1	Check the functionality of creating property markers on the map when entering all criteria with single or multiple values and entering the minimum price value less than the maximum price value	A marker for each filtered property will be created in its correct location on the map.	As expected	RQ1, RQ4, and RQ6
2	Update the criteria by adding one or more values of neighbourhood or property type or the number of bedrooms or increasing the maximum price.	New markers will be added to the map in the correct locations.	As expected	
3	Update the criteria by removing one or more values from the neighbourhood, property type, number of bedrooms or decreasing the maximum price.	The map will be updated by removing markers associated with removed neighbourhoods or property type or the number of bedrooms or price.	As expected	
4	Keep one or more fields of criteria empty and select values for other criteria.	The map will be displayed without any marker.	As expected	

5	Select all property criteria and insert the minimum price greater than the maximum price.	The application will throw an error message that says " The minimum price should be less than the maximum price" and the map will be displayed without any marker.	As expected	
6	Select all property criteria and insert the minimum price with a value less than the maximum but the criteria do not match any property in the database. For example, neighbourhood = Greenwich, Property type = private room, number of bedrooms = 1, minimum price = 10, and maximum price = 100.	The map will be displayed without any marker.	As expected	
Test Embassy functionality				
7	Select a country name.	The application will display contact information for the selected country embassy.	As expected	RQ5
8	Remove the selected country name and keep the field empty.	The application will not return any data.	As expected	
Test Property page				
9	Click on the Listing link in the marker pop-up.	The property page will open in a new tab. This page contains information about the property (name, its page on the Airbnb website, location, neighborhood overview, about host, review score, bar chart of review types) and allows the user to select nearby places.	As expected	RQ1, RQ2, RQ3, and RQ8
Test Food venue pages (Breakfast, Lunch, Dinner, Café)				
10	Click on one of the food venues buttons (Breakfast, Lunch, Dinner, Cafe) on the property page. Then, select a diet type (vegetarian, vegan, gluten-free) from the drop-down list.	The system will display all matches places, ordering them from nearest to farthest. If there are no matching places the application will display a message that says "Sorry, there is no matching place!".	As expected	RQ2, and RQ7

11	Click on one of the food venues buttons (Breakfast, Lunch, Dinner, Cafe) on the property page. Then select a diet type (any type) from the drop-down list.	The application will display all the nearby food venues that filtered based on the meal only (breakfast, lunch, dinner or coffee) and order them from nearest to farthest. If there are no nearby places, the application will display a message that says "Sorry, there is no matching place!".	As expected	
Test Activity venues page				
12	Click on the activity button on the property page.	The application will display all the nearby activity places and order them from nearest to farthest. If there is no activity venue nearby, the application will display a message that says "Sorry, there is no Activity nearby!".	As expected	RQ2, and RQ7
Test Train stations page				
13	Click on the train button on the property page.	The application will display all the nearby train stations and order them from nearest to farthest. If there is no train station nearby, the application will display a message that says "Sorry, there is no Train station nearby!".	As expected	RQ3, and RQ7
Test Bus stops page				
14	Click on the bus button on the property page.	The application will display all the nearby bus stops and order them from nearest to farthest. If there is no bus stop nearby, the application will display a message that says "Sorry, there is no Bus Stop nearby!".	As expected	RQ3, and RQ7

Table 4.1: Test Cases

From the table 4.1 it can clearly be seen that all the functions work as expected. This means that the application meets all the functional requirements in Section 2.3.4 and the reliability requirement from the non-functional requirements in Section 2.3.3. However, to check if the application can meet the usability requirement, which means the user can easily use the application without prior learning, the user evaluation was conducted in Section 4.3.

4.2 Page Speed Testing

This section will investigate the latency of each page of the Travel Guide application. The page speed is one of the essential aspects of web applications, as 47% of the users expect the page load within 2 seconds. Therefore, the average load time will be calculated[27]. The best practice of average page load time is under 3 seconds [28]. However, the average speed of load travel websites is 6.7 seconds, thus, it will be used as a baseline to compare it with the performance of the Travel Guide [28].

As mentioned in 3.2, this application uses Dash library, and it provides data about the page load time in millisecond (ms) in a graph called Callback graph. Not only that, but it also shows the data that passed between the application and the web browser (inputs and outputs) as shown in figure 4.1. The value boxes indicate user inputs, while data is the data that pass from one page to another. To illustrate, the property location (longitude and latitude) is passed via an object called property_data from the property page to the food venue page, and then this data will be used in the Places API to retrieve the nearby places. However, the developer calculated the average load time of 10 times running each page, passing different values and data. Then, summed all the load time in the green boxes, if there is data passed to multiple callbacks, then use these total values to divide by 10. (See table 4.2)

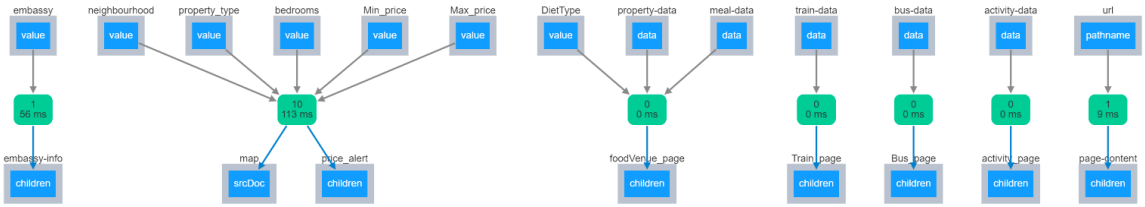


Figure 4.1: Callback graph

From the table 4.2 below, it can be clearly seen that most of the Travel Guide pages have average load times less than one second and are significantly less than the baseline average load time (6.7s). That means the user can update the map and navigate through the pages smoothly and in a reasonable time.

However, the activity page, and the food venue pages, when the user enters the diet type, have a longer page load time. The activity page has an average load time of 1.09 s that because the activities callback section in the code computes the similarity between activity venues that come from Places API and DBpedia API online. However, the average page load time is faster than the user expectation and the baseline. On the other hand, the food venue page average load time is 14.4, which is much slower than the baseline, and the user may leave the website when it takes too long to load the data. The reason for the slowness of page load is that the data comes from two different sources, and the linking process between them is done online as discussed in Section 3.3.2 and 3.4.3.

ID	Test case	Average Load Time(ms)	Average Load Time(s)
1	Open the Map page	295 ms	0.295 s
2	Update the map with one or more value for each criteria (neighbourhood, property type, number of bedrooms) and price range.	144.7 ms	0.1447 s
3	Clear all the criteria to update the map.	518.6 ms	0.5186 s
4	Select a Country name from the Embassy list.	47.7 ms	0.0477 s
5	Clear the country field to update the Embassy section.	28.3 ms	0.0283 s
6	Click on the Listing link to navigate to the property page.	485.3 ms	0.4853 s
7	Navigate to the food venue, by clicking on one of the food venue buttons (Breakfast, Lunch, Dinner or Cafe)	398 ms	0.398 s
8	Select a diet type from the drop-down list	14426 ms	14.426 s
9	Open Activity page	1,090 ms	1.09 s
10	Open Train page	540 ms	0.54 s
11	Open Bus page	534 ms	0.534 s

Table 4.2: Page Speed Testing

4.3 User Evaluation

The user evaluation follows the strategy of Cognitive Walkthroughs to evaluate the product by giving the evaluators a set of tasks, which are the test cases in table 4.1, that help them to explore all the features and evaluate it from their perspective [29]. After exploring the application, the participants were asked to answer an evaluation questionnaire that contains nine questions, eight of them are multiple choices questions (required to be answered) and one is an open-ended question (not required to be answered) this question is used to help the developer to understand the issues that occurred to participants [30]. This strategy is used to reveal issues related to user experience and product usability.

The questioner responses were collected from 10 participants, as mentioned, the users were asked to complete the questioner when they finish exploring the application. These questions allow the participants to measure the usability aspects by using the 5-points Likert scale, which is a tool that measures the product usability with a small number of participants [31].

1. **How easy was it to use the Travel Guide app?**:
   Options: *Extremely easy, Very easy, Moderately easy, Slightly easy, Not at all easy*.
   Outcome: 60% of the participants found the application is very easy to use, while others (40%) rate it as moderately easy.

2. **How user-friendly is the Travel Guide's interface?**:
   Options: *Extremely user-friendly, Very user-friendly, Moderately user-friendly, Slightly user-friendly, Not at all user-friendly*.
   Outcome: 30% of the participants measure the user-friend aspect with "very user friendly", and 70% of them found it "moderately user-friendly".

3. **How often does the Travel Guide app crash?**:
   Options: *Extremely often, Very often, Moderately often, Slightly often, Not at all often*.
   Outcome: 9 out of 10 of the participants (90%), the application did not crash with them and their answer is "Not at all often". On the other hand, one participant (10%) answered it with "Slightly often".

4. **Overall, are you satisfied with the performance of the Travel Guide app?**:
   Options: *Extremely satisfied, Very satisfied, Moderately satisfied, Slightly satisfied, Not at all satisfied*.
   Outcome: 60% of the participants were very satisfied with the performance and 40% of them moderately satisfied with it.

5. **Do the colour-coded map markers help you to find a place to stay in a safe area?**:
   Options: *Extremely helpful, Very helpful, Moderately helpful, Slightly helpful, Not at all helpful*.
   Outcome: 70% found the colour of the markers "very helpful" to find a safe place to stay, and 30% found it "moderately helpful".

6. **I found the bar chart is informative in terms to know customers' opinions.**:
   Options: *Strongly agree, Agree,Neither agree or disagree, Disagree, Strongly disagree.*.
   Outcome: Most of the participants found it informative. 80% of them were strongly agreed and 20% agreed.

7. **The Travel Guide app will save me time when I search for a place to stay.**:
   Options: *Strongly agree, Agree,Neither agree or disagree, Disagree, Strongly disagree.*.
   Outcome: 90% of participants are strongly agreed and 10% agreed that the application will save them time while they are searching for a place to stay.

8. **How likely are you to recommend the Travel Guide app to others?** :
   Options: *Extremely likely, Very likely, Moderately likely, Slightly likely, Not at all likely*.
   Outcome: 60% are very likely, 30% are moderately likely and 10% is slightly likely to recommend the application to others.

The participants had one open-ended question; it was not required to answer ,but allows them to suggest improvements from their perspectives. The question is "From your perspective, how can we improve the Travel Guide app?". 8 out of 10 answered this question, one of them was very optimistic his/her answer is "No need to improve it is very good". 40% of participants appreciated the features of the Travel Guide application. Specifically, presenting the crime rate associated with property markers, the bar chart that provides a summary of customers' reviews, and the suggested nearby places and transportation.

On the other hand, participants suggested improvements for nearby places, by combining all meals and café buttons into one button and another suggestion was to add photos for nearby venues. Moreover. They suggested an improvement for the embassy feature by adding more embassies in the country list. Adding (help me or contact me) feature in case they encounter any issue, weather

forecasting feature, a message on screen in case there are not any available accommodations exact to the user choices, customers' reviews in the marker pop-up, adding an interactive map to navigate to the nearby train stations or bus stops. In addition, one suggestion was to limit the choices of property types, moreover,making the application works faster.

In summary, The results of the user evaluation questionnaire clearly show that the project is easy to use and reliable, successfully built the requirements in Section 2.3.4 and 2.3.3 and solved the shortage information issue. The users found the marker colours help them to find a place to stay in a safe area, and the summary of customers' reviews was informative in terms to have an overlook about the property. Moreover, the users like nearby venues and transportation features. However, even though the embassy function works correctly but it needs more data of all the embassies in London.

Chapter 5

Conclusion

5.1 Project Status

The purpose of this project was to fill the gap found in the current sharing economy applications such as, Airbnb and other platforms that provide places to stay, as discussed in 2.2. The gap was a shortage of information about the activities and places around the properties, the neighbourhood's crime rate, and information about valuable locations like embassies. Providing the user with more effective information will support his/her decision of where he/she can stay in London. Thus, the Travel Guide application relies heavily on data. Therefore, the data collection and extraction process was the essential part of the project, and this process was completed in Colab due to the enormous size of the raw data files. Based on the project plan, the provided properties by Airbnb are presented on an interactive map to let the user set the criteria to visualize the property location. The crime rate is translated into three marker colours that indicate the crime rate in each neighbourhood in London, where green markers indicate that the neighbourhood is safe and has a low crime rate, orange markers indicate the middle level of crime rate, while the red markers indicate a high crime rate. The map markers pass valuable information to the user, not only the indication of crime rate but also provide pop-ups that contain summary information about the property (property image, name, accommodates, number of bedrooms, price and the property page link).

The property page was built to provide the user with detailed information about the property and add a new feature summarising the customers' reviews. This feature was developed by applying the sentiment analysis to the reviews presented in text format. Not only that but also the property page provides the users with different controls (buttons) that allow the user to navigate to the next page, which is nearby venues. Venue pages are generally divided into three categories food, activities and transportation.

The first category, food venues (Breakfast, Lunch, Dinner and Cafe), pages under this category give the user more control in selecting the diet types (Vegetarian, Vegan, Gluten-free or any type). Food venue pages will provide the user with information about all the nearby venues ordered by the distance between the venue and property (nearest to farthest), moreover, they are filtered based on the diet type. The second category is activity venues. This page focuses on providing the user with historical information about activity venues around the property and its address and the distance. The last category is transportation, pages under this category are divided into two types of transportation: train and bus. The train page lets the user know about the nearby train stations, their names, address, distances and lines they serve. On the other hand, the bus page shows the nearby bus stops with their names, address and distances. To illustrate, all the pages order the results from the nearest to the farthest.

Lastly, the Embassy function is provided on the main page (map page) that helps the user to collect contact information of his/her country's embassy in London. This function aims to enrich the user with more information to stay safe in case he/she encounter any issue in London.

In terms of the user interface, the evaluators commended the product user-friendliness and ease of use. The system does not require the user to learn something before using or recalling information. Functionality shows the robustness of the system in terms of updating the information based on the user entries.

5.2 Future Work

The Travel Guide is a web application that relies on the data in order to support the user in selecting a place to stay. Due to the huge size of the current database and shortage of local memory space, the system does not store the data retrieved from Places API and DBpedia API in the database, these data are used to enrich the user with information about nearby food and activity venues. Thus, future work should aim to speed up the retrieving data process by storing them in the database and calculating the similarity between the data from different sources by using Locality-Sensitive Hashing instead of FuzzyWuzzy. Because the Locality-Sensitive Hashing algorithm reduces the data dimensionality [32].

To solve the issues of the few embassies and provide embassies location visualization on the map. Future works would find a source that allows the developer to extract more data or integrate the system with embassies databases (if possible). Moreover, due to the time limitation, the embassies' locations were not visualised on the map. Future work could aim to add new marker colours or shapes to visualize embassies' locations on the map.

Future work could focus on enhancing the visualizing of crime rates on the map with the choropleth layer instead of marker colour. The data of crime rate per month per neighbourhood is stored in the database as mentioned in Section 3.3.2.

The application could be enhanced by adding new features. Due to the scope of this project, it was not possible to add features that are mentioned in Section 2.3.4 that states to recommend a host to the user based on matched interests and predict the property price in the near future. Moreover, the map could be enhanced by zooming in when the property markers are visualised and show a message if the user's criteria do not match any property in the database. In addition, some features are suggested by evaluators, such as adding contact us to allow the user to contact the support team in case the user has questions, adding a navigatable map that allows the user to check the route of the nearby places. Furthermore, adding images for the nearby places, adding customers' reviews summary to the marker pop-ups. Lastly, future works could include a weather forecasting feature.

5.3 Summary

To summarize, the sharing economy platforms in the hospitality field have been used to help the user to find a place to stay. As this field is growing and increasing the number of platforms, users need to have information about the places to stay and the surrounding area. This project aspired to

contribute to this field, by providing a new informative product that supports the user decision to find a place to stay.

Bibliography

[1] Byers J.W. Zervas G. Proserpio D. "The rise of the sharing economy: Estimating the impact of Airbnb on the hotel industry". In: *Journal of marketing research* 54.5 (2017), pp. 687–705.

[2] Martucci B. *What Is the Sharing Economy – Example Companies, Definition, Pros & Cons*. https://www.moneycrashers.com/sharing-economy/. Accessed: 2021-10-15. 2018.

[3] FOLGER J. *Airbnb Advantages and Disadvantages*. https://www.investopedia.com/articles/personal-finance/032814/pros-and-cons-using-airbnb.asp. Accessed: 2021-10-15. 2021.

[4] Airbnb. *What is Airbnb and how does it work?* https://www.airbnb.co.uk/help/article/2503/what-is-airbnb-and-how-does-it-work. Accessed: 2021-10-15.

[5] Ravi S. Yaraghi N. "The Current and Future State of the Sharing Economy". In: *Brookings India* (2017).

[6] *Airbnb Revenue 2018-2021 — ABNB*. https://www.macrotrends.net/stocks/charts/ABNB/airbnb/revenue. Accessed: 2021-10-17.

[7] Gomez M. Mody M. *Airbnb and the Hotel Industry: The Past, Present, and Future of Sales, Marketing, Branding, and Revenue Management*. https://rb.gy/tpgcus. Accessed: 2021-10-17.

[8] Önder I. Gunter U. "Determinants of Airbnb demand in Vienna and their implications for the traditional accommodation industry". In: *Journal of marketing research* 24.3 (2018), pp. 270–293.

[9] Airbnb. *Your safety is our priority*. https://www.airbnb.co.uk/trust. Accessed: 2021-10-15. 2018.

[10] Tabari S Lu L. "Impact of Airbnb on customers' behaviour in the UK hotel industry". In: *Journal of marketing research* 24.1 (2019), pp. 13–26.

[11] Reeves P. Gallagher A. Dunleavy J. *The Waterfall Model: Advantages, disadvantages, and when you should use it*. https://developer.ibm.com/articles/waterfall-model-advantages-disadvantages/. Accessed: 2021-10-19. 2019.

[12] University of Cape Town Computer Science Department. *Context diagrams, Chapter 6. Data-Flow Diagrams*. https://www.cs.uct.ac.za/mit_notes/software/htmls/ch06s06.html. Accessed: 2021-10-21. 2011.

[13] Pykes K. *Vectorization in Python*. https://towardsdatascience.com/vectorization-in-python-46486819d3a. Accessed: 2021-11-9. 2020.

[14] Bahaieva O. *Top 7 Reasons Why You Need to Learn Python as a Data Scientist*. https://towardsdatascience.com/top-10-reasons-why-you-need-to-learn-python-as-a-data-scientist-e3d26539ec00. Accessed: 2021-10-4. 2020.

[15] Google one. *Upgrade to a plan that works for you*. https://one.google.com/about. Accessed: 2021-11-10.

[16] Jet Brains. *The Python IDE for Professional Developers*. https://www.jetbrains.com/pycharm/. Accessed: 2021-9-22.

[17] Beri A. *SENTIMENTAL ANALYSIS USING VADER*. https://towardsdatascience.com/sentimental-analysis-using-vader-a3415fef7664. Accessed: 2021-11-10. 2020.

[18] *Basic Dash Callbacks*. https://dash.plotly.com/basic-callbacks. Accessed: 2021-10-9.

[19] W3schools. *What is Bootstrap?* https://www.w3schools.com/whatis/whatis_bootstrap.asp. Accessed: 2021-11-11.

[20] Foursquare. *Places API rate-limits*. https://developer.foursquare.com/docs/places-api/rate-limits/. Accessed: 2021-10-10.

[21] Pypi. *fuzzywuzzy 0.18.0*. https://pypi.org/project/fuzzywuzzy/. Accessed: 2021-11-20.

[22] Souissi N. El Arass M. "Data Lifecycle: From Big Data to SmartData". In: *2018 IEEE 5th International Congress on Information Science and Technology (CiSt)*. 2018, pp. 80–87. DOI: 10.1109/CIST.2018.8596547.

[23] Tripadviser. *About Tripadvisor*. https://tripadvisor.mediaroom.com/uk-about-us. Accessed: 2021-10-21. 2019.

[24] *Google Maps Places APIs Overview*. https://developers.google.com/maps/documentation/places/web-service/overview. Accessed: 2021-10-21.

[25] *Pricing that scales to fit your needs*. https://mapsplatform.google.com/pricing/. Accessed: 2021-10-22.

[26] *Interquartile Range*. https://stattrek.com/statistics/dictionary.aspx?definition=interquartile%20range. Accessed: 2021-10-7.

[27] *How to Analyze and Improve Page Load Performance*. https://www.pingdom.com/blog/how-to-analyze-and-improve-page-load-performance/. Accessed: 2021-12-8. 2018.

[28] Anderson S. *How fast should a website load in 2022?* https://www.hobo-web.co.uk/your-website-design-should-load-in-4-seconds/. Accessed:2021-12-8. 2021.

[29] Conyer M. "User and usability testing-how it should be undertaken?" In: *Australasian Journal of Educational Technology* 11.2 (1995).

[30] Veal R. *How To Write Effective Usability Testing Questions: A Beginner's Guide*. https://careerfoundry.com/en/blog/ux-design/how-to-write-usability-testing-questions/. Accessed: 2021-11-25. 2021.

[31] Usability.gov. *System Usability Scale (SUS)*. https://www.usability.gov/. Accessed: 2021-11-22.

[32] Mclendon R. *Building a Recommendation Engine with Locality-Sensitive Hashing (LSH) in Python*. https://www.learndatasci.com/tutorials/building-recommendation-engine-locality-sensitive-hashing-lsh-python/. Accessed:2021-12-1.

Appendix A

Map page

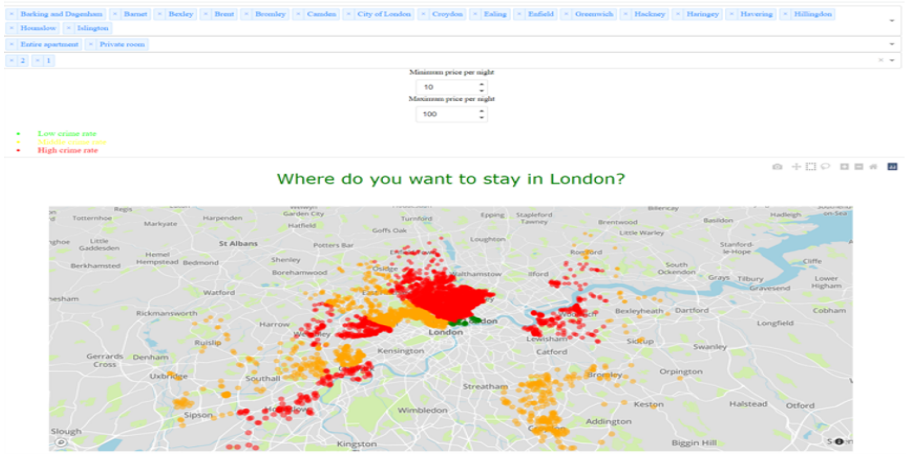


Figure A.1: The first version of the map page.

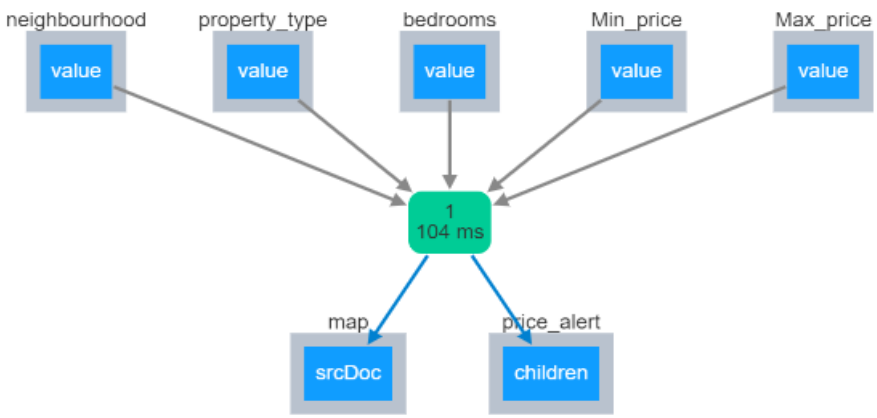


Figure A.2: Map page Callback function I/O

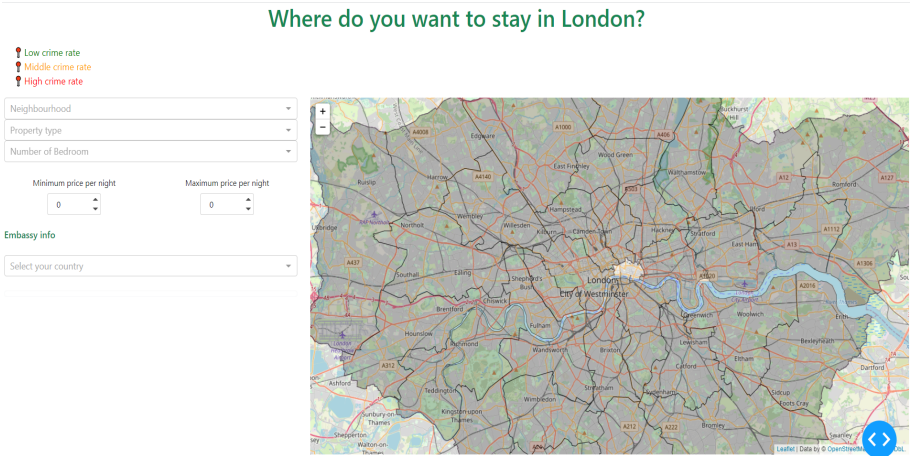


Figure A.3: Map page.

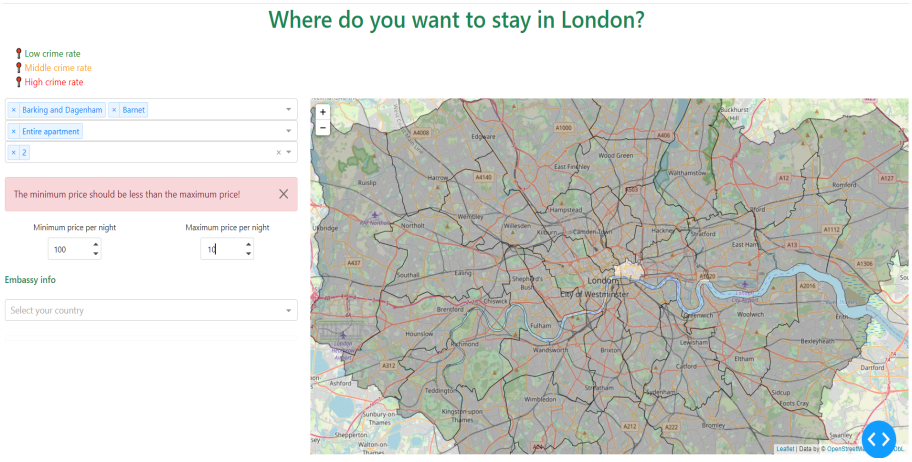


Figure A.4: Map page with price error

Figure A.5: Embassy Callback I/O

Appendix B

Venue and Transportation

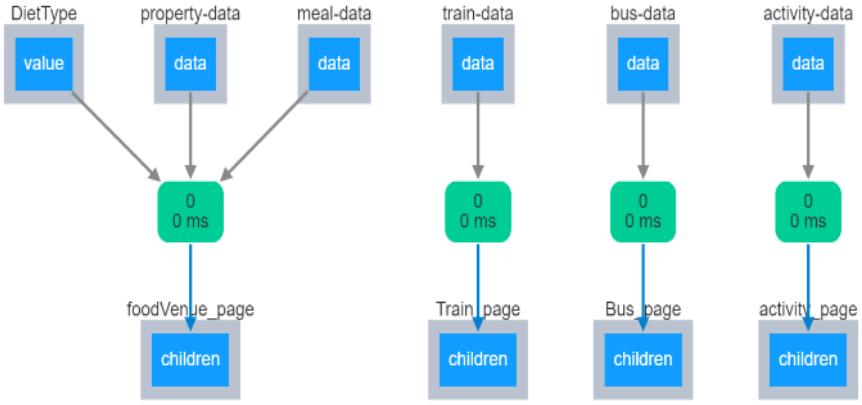


Figure B.1: Venue and Transportation callbacks I/O

Appendix C

Database Schema

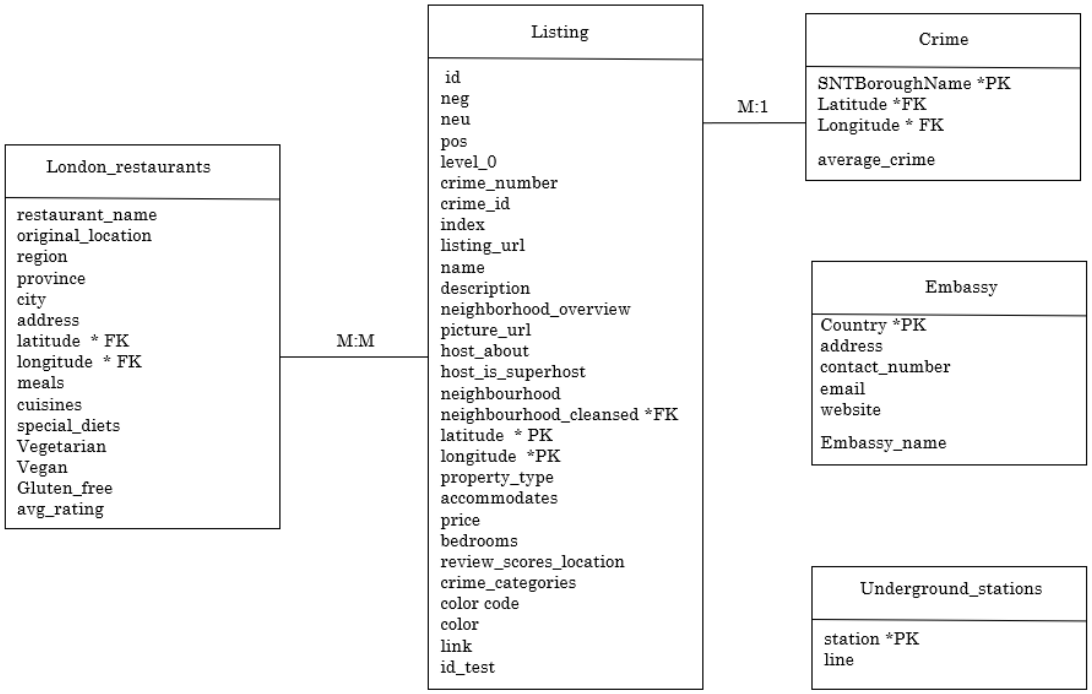


Figure C.1: Entity relationship diagram.

Appendix D

Storyboard

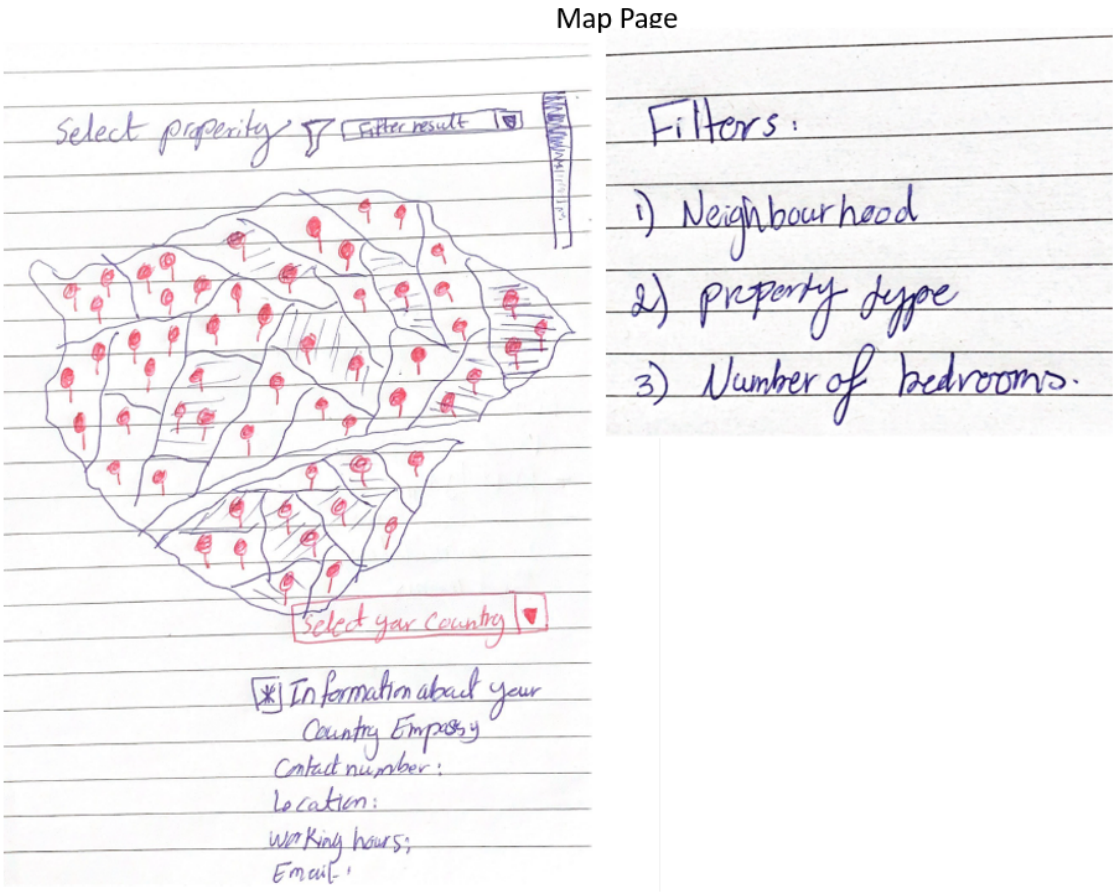


Figure D.1: Storyboard, map page.

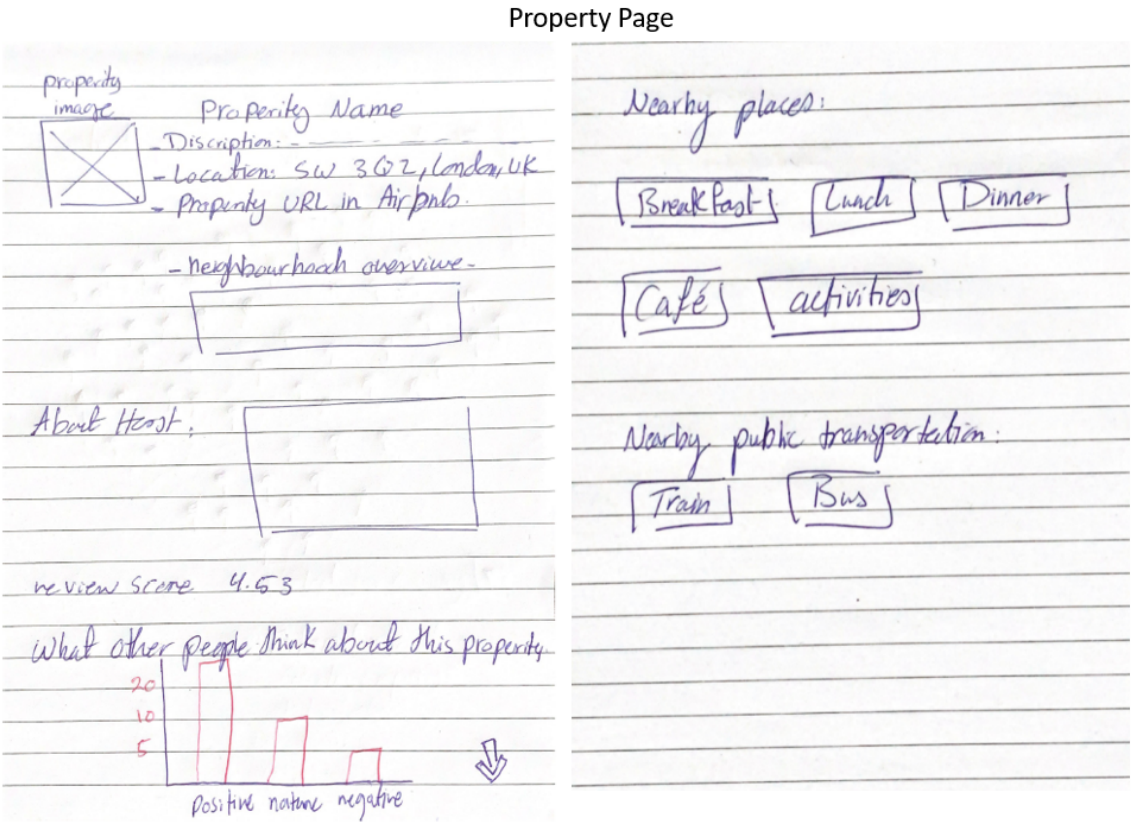


Figure D.2: Storyboard, property page.

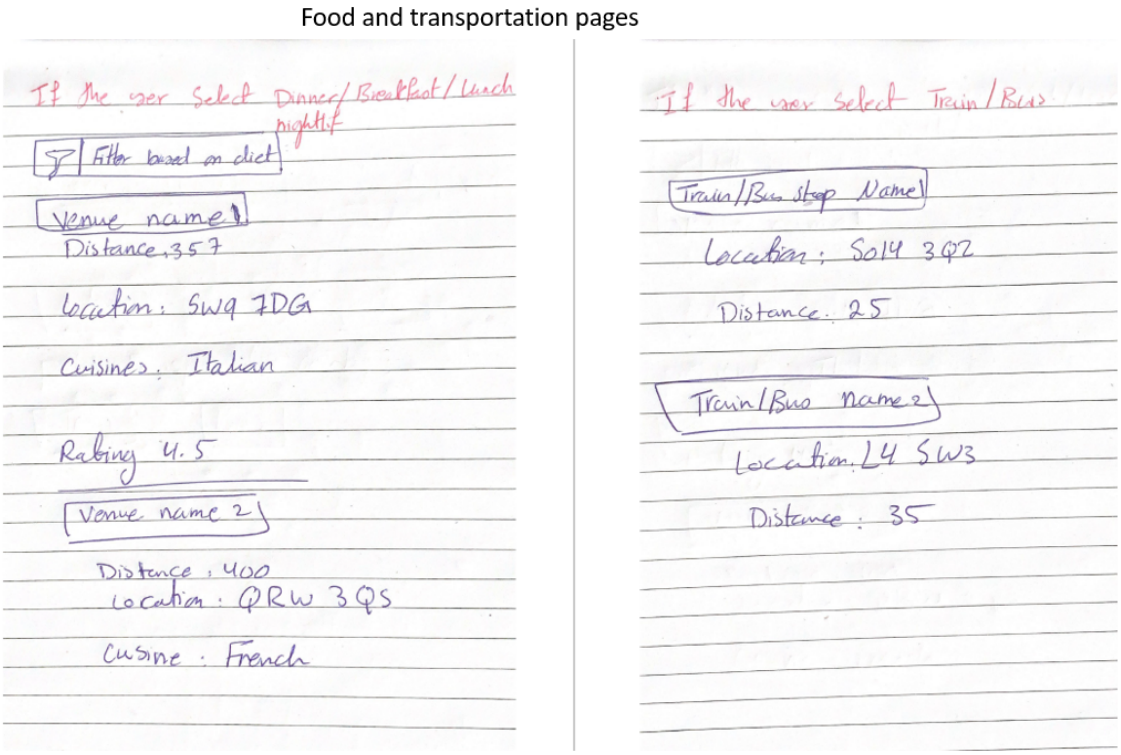


Figure D.3: Storyboard, food and transportation pages.

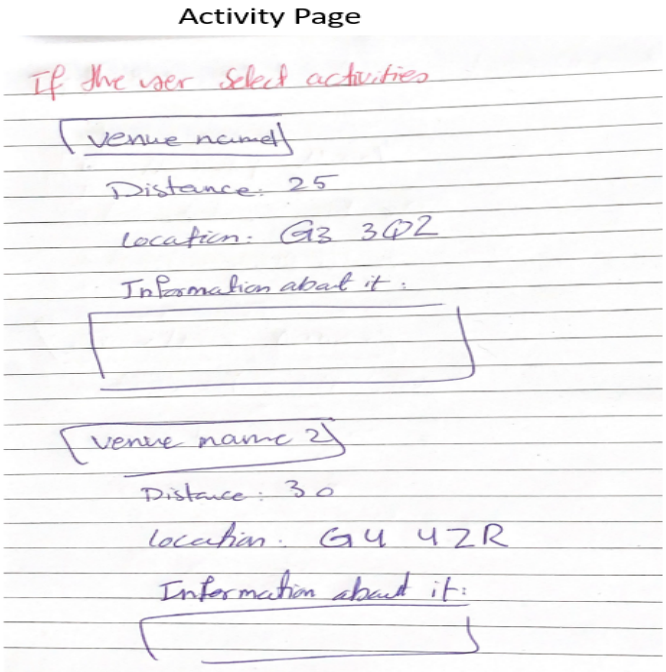


Figure D.4: Storyboard, activity page.

Appendix E

User Evaluation

School of Computing Science
University of Glasgow

Ethics checklist form for 3rd/4th/5th year, and taught MSc projects

This form is only applicable for projects that use other people ('participants') for the collection of information, typically in getting comments about a system or a system design, getting information about how a system could be used, or evaluating a working system.

If no other people have been involved in the collection of information, then you do not need to complete this form.

If your evaluation does not comply with any one or more of the points below, please contact the Chair of the School of Computing Science Ethics Committee (matthew.chalmers@glasgow.ac.uk) for advice.

If your evaluation does comply with all the points below, please sign this form and submit it with your project.

---

1. Participants were not exposed to any risks greater than those encountered in their normal working life.

   Investigators have a responsibility to protect participants from physical and mental harm during the investigation. The risk of harm must be no greater than in ordinary life. Areas of potential risk that require ethical approval include, but are not limited to, investigations that occur outside usual laboratory areas, or that require participant mobility (e.g. walking, running, use of public transport), unusual or repetitive activity or movement, that use sensory deprivation (e.g. ear plugs or blindfolds), bright or flashing lights, loud or disorienting noises, smell, taste, vibration, or force feedback

2. The experimental materials were paper-based, or comprised software running on standard hardware.

   Participants should not be exposed to any risks associated with the use of non-standard equipment: anything other than pen-and-paper, standard PCs, laptops, iPads, mobile phones and common hand-held devices is considered non-standard.

3. All participants explicitly stated that they agreed to take part, and that their data could be used in the project.

   If the results of the evaluation are likely to be used beyond the term of the project (for example, the software is to be deployed, or the data is to be published), then signed consent is necessary. A separate consent form should be signed by each participant.

   Otherwise, verbal consent is sufficient, and should be explicitly requested in the introductory script.

4. No incentives were offered to the participants.

   The payment of participants must not be used to induce them to risk harm beyond that which they risk without payment in their normal lifestyle.

Figure E.1: Ethics checklist form.

5. No information about the evaluation or materials was intentionally withheld from the participants.
   Withholding information or misleading participants is unacceptable if participants are likely to object or show unease when debriefed.

6. No participant was under the age of 16.
   Parental consent is required for participants under the age of 16.

7. No participant has an impairment that may limit their understanding or communication.
   Additional consent is required for participants with impairments.

8. Neither I nor my supervisor is in a position of authority or influence over any of the participants.
   A position of authority or influence over any participant must not be allowed to pressurise participants to take part in, or remain in, any experiment.

9. All participants were informed that they could withdraw at any time.
   All participants have the right to withdraw at any time during the investigation. They should be told this in the introductory script.

10. All participants have been informed of my contact details.
    All participants must be able to contact the investigator after the investigation. They should be given the details of both student and module co-ordinator or supervisor as part of the debriefing.

11. The evaluation was discussed with all the participants at the end of the session, and all participants had the opportunity to ask questions.
    The student must provide the participants with sufficient information in the debriefing to enable them to understand the nature of the investigation. In cases where remote participants may withdraw from the experiment early and it is not possible to debrief them, the fact that doing so will result in their not being debriefed should be mentioned in the introductory text.

12. All the data collected from the participants is stored in an anonymous form.
    All participant data (hard-copy and soft-copy) should be stored securely, and in anonymous form.

Project title Travel Guid System

Student's Name Dina Alsharif

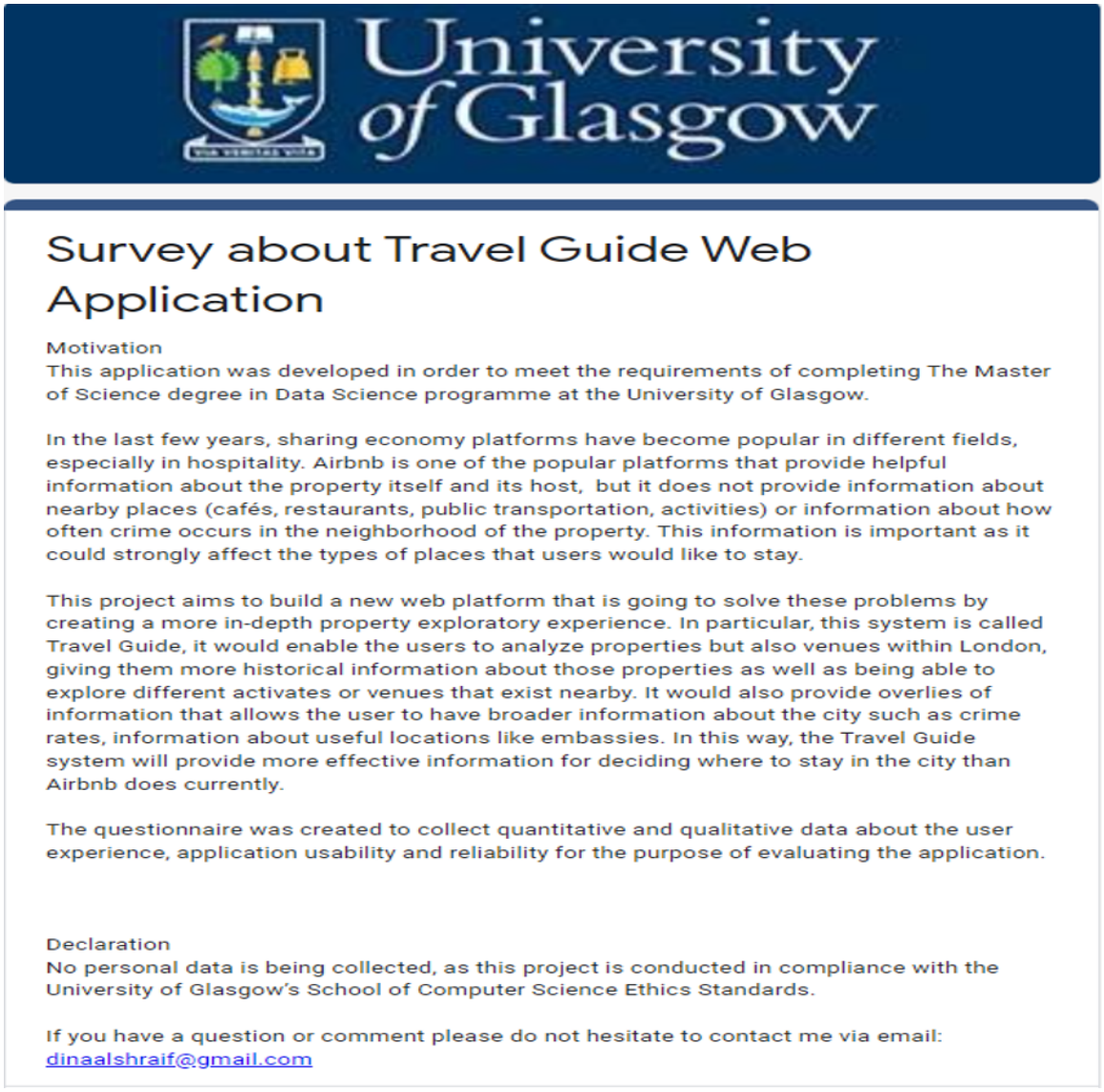
Student Number 2593479A

Student's Signature

Supervisor's Signature Richard McCreadie

Date 23/11/2021

Figure E.2: Ethics checklist form.



University of Glasgow

Survey about Travel Guide Web Application

Motivation
This application was developed in order to meet the requirements of completing The Master of Science degree in Data Science programme at the University of Glasgow.

In the last few years, sharing economy platforms have become popular in different fields, especially in hospitality. Airbnb is one of the popular platforms that provide helpful information about the property itself and its host, but it does not provide information about nearby places (cafés, restaurants, public transportation, activities) or information about how often crime occurs in the neighborhood of the property. This information is important as it could strongly affect the types of places that users would like to stay.

This project aims to build a new web platform that is going to solve these problems by creating a more in-depth property exploratory experience. In particular, this system is called Travel Guide, it would enable the users to analyze properties but also venues within London, giving them more historical information about those properties as well as being able to explore different activates or venues that exist nearby. It would also provide overlies of information that allows the user to have broader information about the city such as crime rates, information about useful locations like embassies. In this way, the Travel Guide system will provide more effective information for deciding where to stay in the city than Airbnb does currently.

The questionnaire was created to collect quantitative and qualitative data about the user experience, application usability and reliability for the purpose of evaluating the application.

Declaration
No personal data is being collected, as this project is conducted in compliance with the University of Glasgow's School of Computer Science Ethics Standards.

If you have a question or comment please do not hesitate to contact me via email:
dinaalshraif@gmail.com

Figure E.3: Travel Guide Survey.

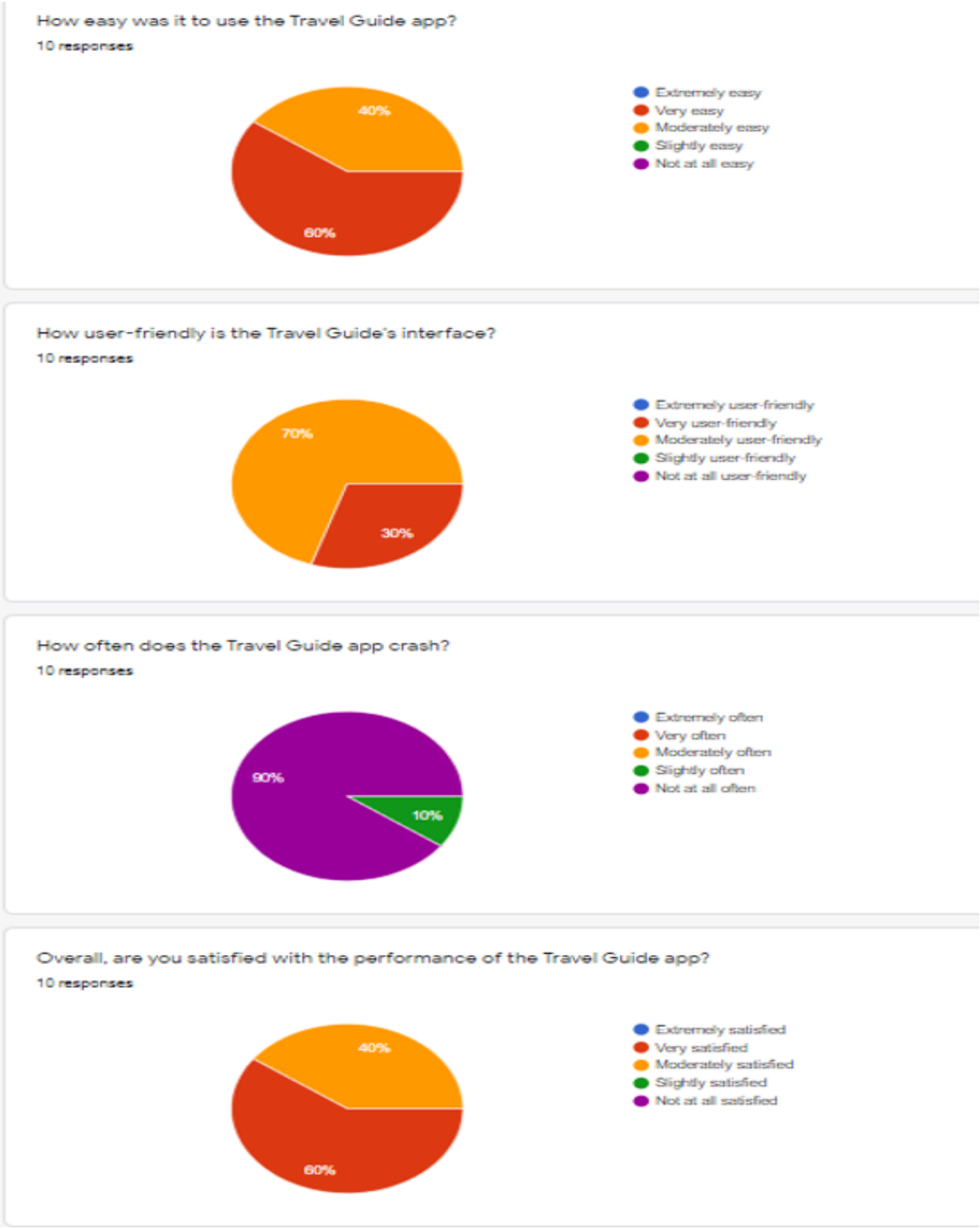


Figure E.4: User evaluation results.

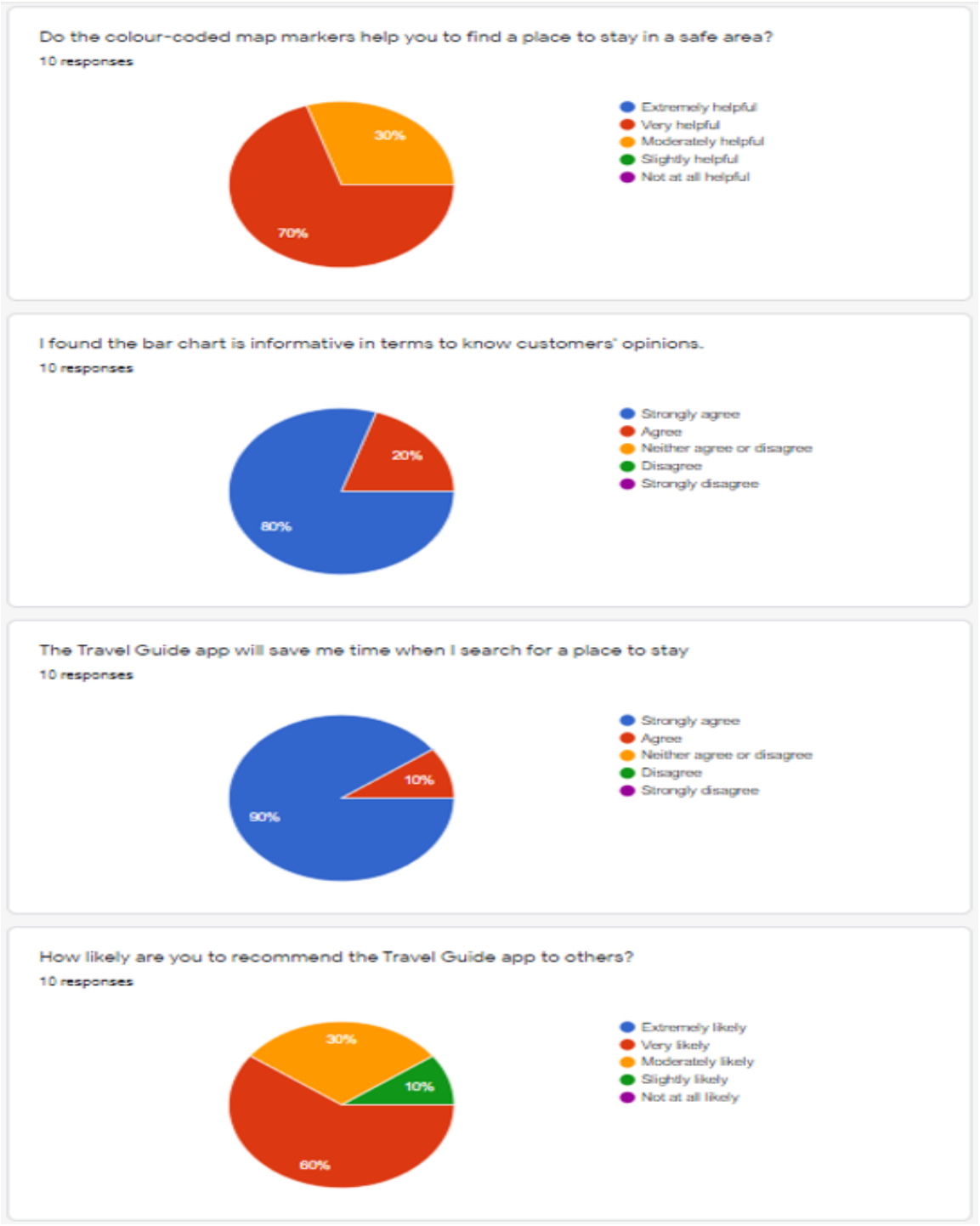


Figure E.5: User evaluation results.

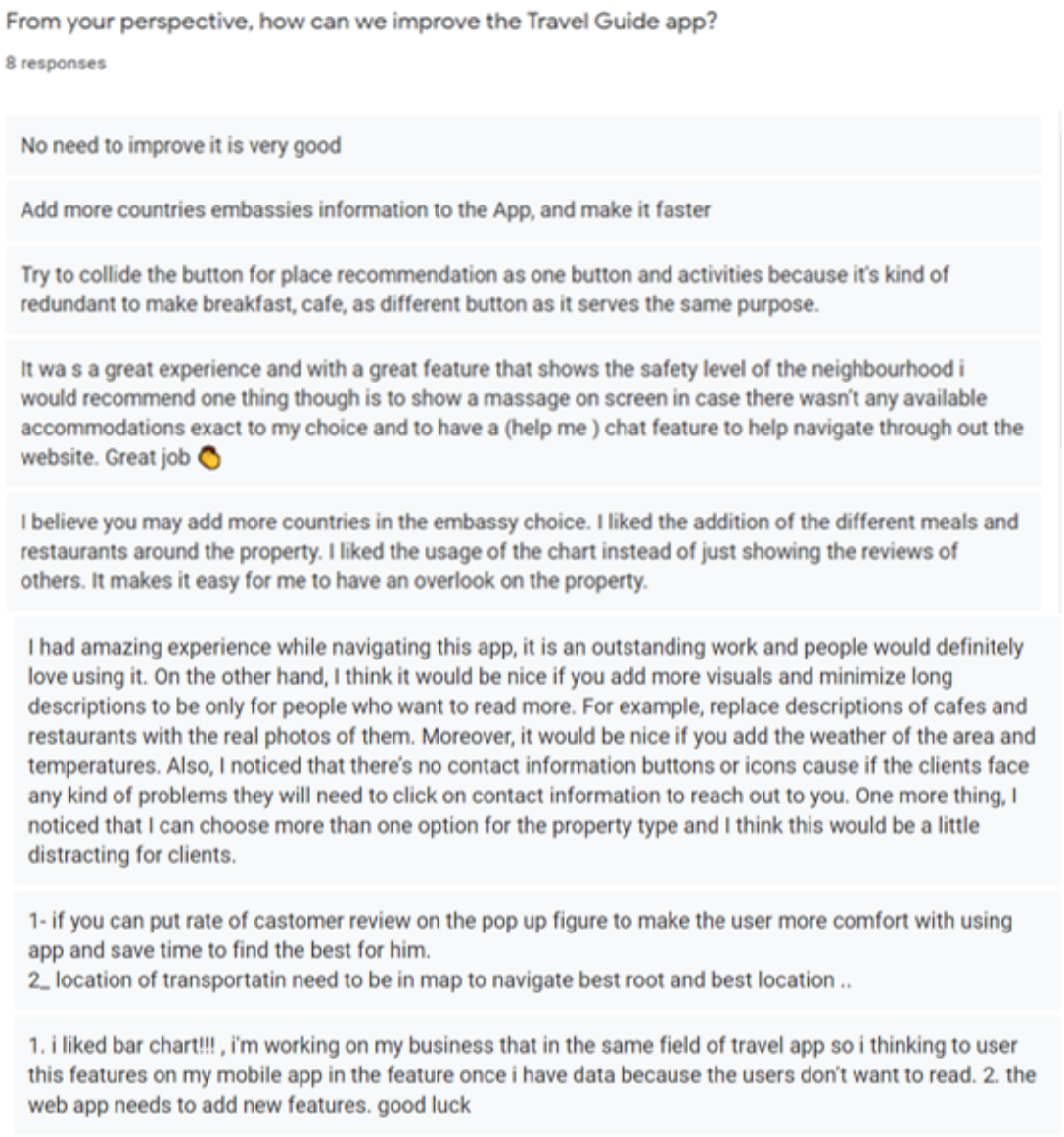


Figure E.6: Open-ended question answers.